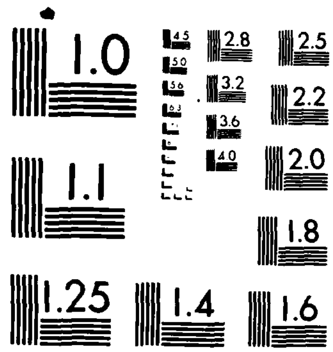
AD-A161 870 A LEAST SQUARES ALGORITHM FOR FITTING PIECEWISE LINEAR FUNCTIONS ON FIXED DOMAINS(U) DEFENCE RESEARCH ESTABLISHMENT ATLANTIC DARTMOUTH (NOVA SCOTI 1/1

UNCLASSIFIED B A TRENHOLM SEP 85 DREA-TM-85/215 F/G 12/1 NL

END



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

UNLIMITED DISTRIBUTION

National Defence
Research and Development Branch

Défense Nationale
Bureau de Recherche et Développment

TECHNICAL MEMORANDUM 85/215

September 1985

A LEAST SQUARES ALGORITHM FOR FITTING PIECEWISE LINEAR FUNCTIONS ON FIXED DOMAINS

B. A. Trenholm

Defence Research Establishment Atlantic

Centre de Recherches pour la Défense Atlantique

Canada

National Defence
Research and Development Branch

Défense Nationale
Bureau de Recherche et Développment

A LEAST SQUARES ALGORITHM FOR FITTING PIECEWISE LINEAR FUNCTIONS ON FIXED DOMAINS

B. A. Trenholm

September 1985

Approved by R.F. Brown   Director/Underwater Acoustics Division

DISTRIBUTION APPROVED BY

R F Brown

D/UAD

TECHNICAL MEMORANDUM 85/215

Defence Research Establishment Atlantic

Centre de Recherches pour la Défense Atlantique

Canada

ABSTRACT

A least squared error algorithm is presented for fitting a piecewise linear function to observed data, when the slope of the function is allowed to change only at specified points. The algorithm can be used to estimate piecewise linear trends in data, where the locations of possible changes in trend are known. Furthermore, it can be used to reduce large quantities of data to manageable sizes, since J coordinates are input and $L \ll J$ coordinates are output. In addition, the algorithm is robust to problems of data dropout provided it is used cautiously. An example of a possible application is the fitting of a linear segment bathythermal profile (temperature vs. depth) to a large quantity of data.

RESUME

On présente un algorithme de correction d'erreur par la méthode des moindres carrés qui permet d'ajuster une fonction linéaire par morceaux à un ensemble de données d'observation, quand la pente de la fonction ne peut varier qu'à des points précis. Cet algorithme peut servir à estimer les tendances linéaires par morceaux des données, à la condition que soient connus les points des variations possibles des tendances. Il peut aussi être employé pour réduire de grands ensembles de données à des ensembles de tailles abordables, puisque pour J coordonnées d'entrée il produit $L \ll J$ coordonnées de sortie. Enfin, l'algorithme est insensible aux problèmes des pertes de données, pourvu qu'on s'en serve avec précaution. Une de ses applications possibles est l'ajustement d'un profil bathythermique linéaire par segments (température en fonction de la profondeur) à un grand ensemble de données.

Contents

1 Introduction

A least squared error algorithm is presented for fitting a piecewise linear function to observed data. The slope of the function is allowed to change only at specified points. This algorithm can be used to estimate piecewise linear trends in data, where the locations of possible changes in trend are known. A possible application in underwater acoustics is the fitting of a linear segment bathythermal profile (temperature vs. depth) to a large quantity of observed data.

1.1 Statement of the Problem

The observed data consist of J coordinate pairs $\{x_j, y_j\}, j = 1, \ldots J$ where x_j is the independent variable and y_j is the dependent variable. The objective is to fit a piecewise linear function to this data. The function is allowed to change slope at $x = u_l$, $l = 1, \ldots L$. The values u_l are given, in monotonic increasing order $u_1 < u_2 < \ldots < u_L$. The fitted curve can be described by its coordinates $\{u_l, v_l\}, l = 1, \ldots L$ where L is much smaller than J. The fitted curve should be one which minimizes the sum of the squared errors between the observed data and the fitted curve, with allowance for possible weighting of the errors. The piecewise linear function $f(x)$ takes the following form:

$$f(x) = v_l + \frac{x - u_l}{u_{l+1} - u_l}[v_{l+1} - v_l] \quad ; \quad u_l \leq x < u_{l+1} \tag{1.1}$$

1.2 The Least Squares Solution

The domain of x is divided into the following L fixed intervals I_l:

$$\begin{aligned} I_1 = &\quad (-\infty, u_1) \cup [u_1\ , u_2) \\ I_2 = &\quad [u_2\ , u_3) \\ &\quad \ldots \\ I_l = &\quad [u_l\ , u_{l+1}) \\ &\quad \ldots \\ I_L = &\quad [u_L\ , \infty) \end{aligned} \tag{1.2}$$

An additional point u_{L+1} is defined as follows:

$$u_{L+1} = u_L + \frac{1}{L-1}[u_L - u_1] \quad . \tag{1.3}$$

The function $f(x)$ is rewritten as a function of a known constant f_0 plus an unknown intercept a_0 plus the unknown function increments $a_l = v_{l+1} - v_l$:

$$f(x) = f_0 + a_0 + \sum_{k=1}^{l(j)-1} a_k \; + a_{l(j)} \left(\frac{x - u_{l(j)}}{u_{l(j)+1} - u_l(j)} \right) \;\; ; \quad x \in I_{l(j)} \, . \tag{1.4}$$

The error ϵ_j associated with the data point $\{x_j, y_j\}$ is defined as the difference between the fitted function $f(x_j)$ and the observed value y_j:

$$\begin{aligned} f(x_j) + \epsilon_j &= \;\; y_j \qquad\quad ; \quad j = 1, \ldots J \\ a_0 + \sum_{k=1}^{l(j)-1} a_k \; + a_{l(j)} \left(\frac{x_j - u_{l(j)}}{u_{l(j)+1} - u_{l(j)}} \right) + \epsilon_j &= \;\; y_j - f_0 \quad ; \quad j = 1, \ldots J \; . \end{aligned} \tag{1.5}$$

This is a system of J linear equations in $L + 1$ unknowns a_l, which may be written in matrix notation as follows:

$$\mathbf{B}_{J \times (L+1)} \mathbf{a}_{(L+1) \times 1} \; + \; \mathbf{e}_{J \times 1} = \mathbf{y}_{J \times 1} \; - \; f_0 \cdot \mathbf{1}_{J \times 1} \tag{1.6}$$

where row j of the matrix $\mathbf{B}$ is as follows:

$$\left[1_{(0)}, \quad 1_{(1)}, \quad \cdots \quad 1_{(l(j)-1)}, \quad \left(\frac{x_j - u_{l(j)}}{u_{l(j)+1} - u_{l(j)}} \right), \quad 0, \ldots, 0 \right] \tag{1.7}$$

and where the vector of unknowns $\mathbf{a}$ is as follows:

$$\mathbf{a}^T = [a_0, a_1, \ldots, a_L] \; . \tag{1.8}$$

The least squares solution $\mathbf{a}$ minimizes $\mathbf{e}^T \mathbf{e} = \sum_{j=1}^{J} \epsilon_j^2$. The solution is obtained by setting the gradient of this function with respect to $\mathbf{a}$ equal to zero:

$$\nabla_{\mathbf{a}} \left(\mathbf{e}^T \mathbf{e} \right) = \mathbf{0} \; . \tag{1.9}$$

This results in the following system of linear equations:

$$[\mathbf{B}^T \mathbf{B}]_{(L+1) \times (L+1)} \mathbf{a}_{(L+1) \times 1} = [\mathbf{B}^T (\mathbf{y} \; - \; f_0 \cdot \mathbf{1})]_{(L+1) \times 1} \; . \tag{1.10}$$

The inverse of the matrix $\mathbf{B}^T\mathbf{B}$ exists if and only if $\mathbf{B}^T\mathbf{B}$ has full rank $L + 1$. This condition is satisfied if the x_j are unique, $J \geq L + 1$ and if each interval I_l contains at least one interior data point x_j. If empty intervals occur, the problem can be decomposed into smaller subproblems where each subproblem contains no empty intervals. In this case $f(x)$ on an empty interval is defined to be the straight line segment joining the end of the last nonempty interval with the beginning of the next nonempty interval.

A weighted least squares solution is one which minimizes $\sum_{j=1}^{J} w_j \epsilon_j^2$ where the w_j are positive weights. The weighted solution is obtained by multiplying row j of matrix $\mathbf{B}$ and the right hand side $y_j - f_0$ by the constant $\sqrt{w_j}$, $j = 1, \ldots J$.

For numerical stability the constant f_0 is chosen to be the mean of the observed values y_j. The condition number of $\mathbf{B}^T\mathbf{B}$ is best if the data points are located near the centers of the intervals.

2 The Least Squares Algorithm

2.1 Initialization

The algorithm begins with a first pass at the observed data $\{x_j, y_j\}$ to compute the mean:

$$f_0 = \overline{y} = \frac{1}{J}\sum_{j=1}^{J} y_j \tag{2.1}$$

and to count the number of data points n_l in each interval I_l, $l = 1, \dots L$.

2.2 Weighted Errors

The error weighting function may be chosen as required, depending on the application. For illustration, it will be assumed that all data points within a given interval should be equally weighted. It will also be assumed that the average error for each nonempty interval should be equally weighted. The following weighting function reflects these assumptions:

$$w_j = \frac{1}{c + n_{l(j)}} \quad ; \quad x_j \in I_{l(j)} \quad , \tag{2.2}$$

where $c \geq 0$ is a constant which may be used to modify the effect of segments which contain very few points. In the examples which follow, c equals 10.

2.3 Subproblem 1

The first subproblem is numbered $n = 1$. The values n_l are scanned to find the first set of nonempty adjoining intervals $\Omega_1 = \{I_{k(1)}, \dots, I_{k(1)-1+L(1)}\}$. Subproblem n has $L(n)$ intervals with $L(n) + 1$ unknowns.

2.3.1 Calculation of the Least Squares Matrix

The coefficients of the least squares system in Equation 1.10 are obtained by processing the data set $\{x_j, y_j\}$ point by point without bringing the entire data set into the computer memory. It is assumed that the data are in random order; therefore the entire data set must be scanned in order to solve a subproblem. The matrix $\mathbf{C} = \mathbf{B}^T\mathbf{B}$ and the right hand side vector $\mathbf{d} = \mathbf{B}^T[\mathbf{y} - f_0 \cdot \mathbf{1}]$ are initially set to zero:

$$\begin{aligned} C_{im} = &\ 0 \ ; \ i = 1, \ldots L(n)+1 \ ; \ m = 1, \ldots L(n)+1 \\ d_i = &\ 0 \ ; \ i = 1, \ldots L(n)+1 \ . \end{aligned} \tag{2.3}$$

The data point $\{x_j, y_j\}$ is contained in interval $I_{l(j)}$. If $I_{l(j)} \not\subset \Omega_n$ the data point is ignored. If $I_{l(j)} \subset \Omega_n$ then the following extended row vector $\mathbf{b}$ is defined with dimension $L(n)+2$:

$$\mathbf{b} = \left[1_{(0)},\ 1_{(1)},\ \ldots\ 1_{(l(j)-k(n))},\ \left(\frac{x_j - u_{l(j)}}{u_{l(j)+1} - u_{l(j)}} \right),\ 0, \ldots, 0,\ y_j - f_0 \right] \ . \tag{2.4}$$

The jth data point then makes the following contribution to $\mathbf{C}$ and $\mathbf{d}$:

$$\begin{aligned} C'_{im} = &\ C_{im} + w_j b_i b_m \qquad ; \ i = 1, \ldots L(n)+1 \ ; m = 1, \ldots L(n)+1 \\ d'_i = &\ d_i \quad + w_j b_i b_{L(n)+2} \ ; \ i = 1, \ldots L(n)+1 \end{aligned} \tag{2.5}$$

where the prime denotes an update to the coefficient values.

2.3.2 Solution of the Least Squares System

The system of linear equations is

$$\mathbf{Ca} = \mathbf{d} \tag{2.6}$$

Since the matrix $\mathbf{C}$ is symmetric, an efficient inversion algorithm could be invoked. However there are cases where $\mathbf{C}$ may be ill-conditioned if the data are sparse and if some of the data points $\{x_j, y_j\}$ are very close together. A modified gaussian elimination technique has been selected instead. This technique is based on the Simplex Method of linear programming [1]. If the matrix $\mathbf{C}$ has rank $M < L(n)+1$, then the algorithm assigns arbitrary values to $L(n)+1-M$ variables. The preferred arbitrary variables are $a_M, \ldots, a_{L(n)}$. Arbitrary default values for all variables should be chosen to suit the application. For bathythermal curve fitting the default values may be set to zero, or they may be chosen based on historical data.

2.3.3 Assignment of Coordinates of f(x)

After solution of the first subproblem $n = 1$ the solution of $f(x)$ is determined for $x \leq u_{k(1)+L(1)}$. The coordinates v_l are calculated as follows:

$$v_l = f(u_l) = \begin{cases} f_0 + a_0 + a_1 \left(\dfrac{u_l - u_{k(1)}}{u_{k(1)+1} - u_{k(1)}} \right) & ; \; l \leq k(1) \\ \\ f_0 + a_0 + \displaystyle\sum_{m=k(1)}^{l-1} a_{1+m-k(1)} & ; \; k(1) < l \leq k(1) + L(1) \; . \end{cases} \tag{2.7}$$

This assumes that if the first intervals contain no data points, the initial coordinates are obtained by linear extrapolation of the first nonempty segment; these coordinates could also be specified arbitrarily, depending on the application.

2.4 Termination Criteria

At the end of subproblem n, one of the following cases will apply:

Case I. If $k(n) + L(n) = L$, then $f(x)$ has been determined for all x, and there are no more subproblems to solve.

Case II. If $k(n) + L(n) < L$, and the remaining intervals are empty, then $f(x)$ may be extrapolated linearly from the last segment:

$$\begin{aligned} v_l = {} & f(u_l) \\ = {} & f_0 + a_0 + \sum_{m=k(n)}^{k(n)+L(n)-1} a_{1+m-k(n)} + a_{L(n)} \left(\frac{u_l - u_{k(n)+L(n)-1}}{u_{k(n)+L(n)} - u_{k(n)+L(n)-1}} \right) \; ; \; l > k(n) + L(n) \; . \end{aligned} \tag{2.8}$$

or the coordinates may be specified arbitrarily. The function $f(x)$ is then completely defined and there are no more subproblems to solve.

Case III. If $k(n) + L(n) < L$ and there are non-empty intervals remaining, i.e.

$$\sum_{l=k(n)+L(n)+1}^{L} n_l \neq 0 \; , \tag{2.9}$$

then at least one more subproblem must be solved.

2.5 Subproblem n

The nth subproblem begins with the identification of the next set of nonempty adjoining intervals $\Omega_n = \{I_{k(n)}, \ldots I_{k(n)-1+L(n)}\}$. The new least squares matrix C and vector d are computed and the system is solved for the new unknowns a. At the end of subproblem n ($n > 1$) the solution of $f(x)$ is determined for

$$u_{k(n)} \leq x \leq u_{k(n)+L(n)} \quad . \tag{2.10}$$

The coordinates v_l are calculated as follows:

$$v_l = f(u_l) = \begin{cases} f_0 + a_0 & ; \quad l = k(n) \\ \\ f_0 + a_0 + \displaystyle\sum_{m=k(n)}^{l-1} a_{1+m-k(n)} & ; \quad k(n) < l \leq k(n) + L(n) \quad . \end{cases} \tag{2.11}$$

The intermediate coordinates between problem $n-1$ and problem n are defined as follows:

$$v_l = v_{k(n-1)+L(n-1)} + \frac{v_{k(n)} - v_{k(n-1)+L(n-1)}}{u_{k(n)} - u_{k(n-1)+L(n-1)}} (u_l - u_{k(n-1)+L(n-1)}) \ ; \ k(n-1)+L(n-1) < l < k(n) \ . \tag{2.12}$$

The algorithm then determines which of the three termination cases applies at the end of subproblem n. If Case I or Case II applies, the algorithm terminates. If Case III applies, another subproblem must be solved.

3 Examples

A few examples have been chosen to demonstrate the curve fitting procedure in extreme cases. The bathythermal data are from shallow water. Appendix A contains a listing of FORTRAN program FITBT which specifies the required values of u_l and default values for the variables a_l. Appendix B contains a listing of FORTRAN subroutine LSQPL which implements the least-squares algorithm.

3.1 Example 1: A Well-Behaved Problem

Figure 3.1 shows a well-behaved example in which there are many data points. The fixed depths u_l are indicated by dashed lines. Since all of the depth intervals contain data points, there is only one problem to solve. Figure 3.2 shows the fitted piecewise linear curve.

3.2 Example 2: Problem Decomposes into Two Subproblems

Figure 3.3 shows an example in which there are several empty intervals. The problem decomposes into two independent subproblems, as shown in Figure 3.4. The intermediate coordinates are defined by a linear segment joining the two independent solutions. The first coordinates are obtained by linear extrapolation of the first nonempty interval. The last coordinates are obtained by linear extrapolation of the last nonempty interval.

3.3 Example 3: An Example with Sparse Data

Figure 3.5 is an extreme example in which the number of unknowns exceeds the number of data points. One solution, shown in Figure 3.6, sets the free variables to zero; that is, the temperature increments in the associated intervals (4 and 10) are zero. A second solution, shown in Figure 3.7, chooses the free variable values based on historical temperature gradient data, Figure 3.8. Appendix C contains the corresponding input and output files for example 3.

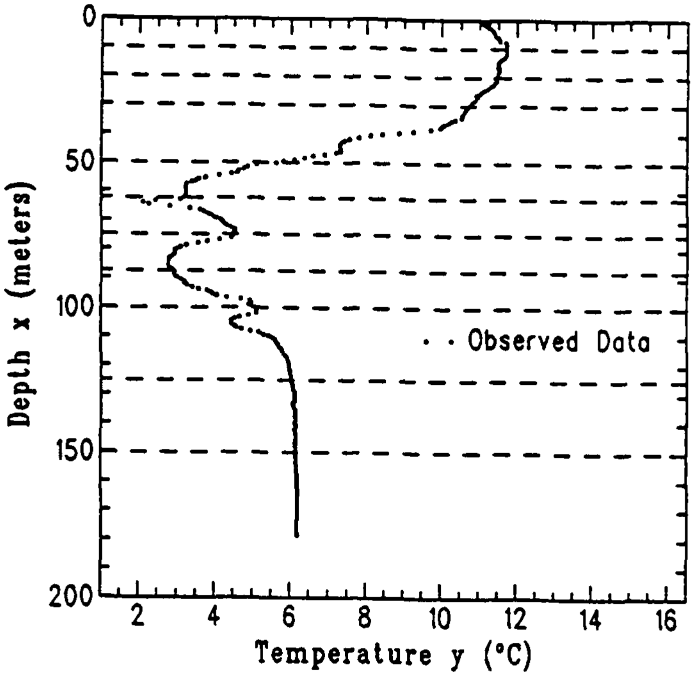


Figure 3.1: Example 1. A well-behaved example with more data points than unknowns. The fixed depths u_l are indicated by dashed lines.

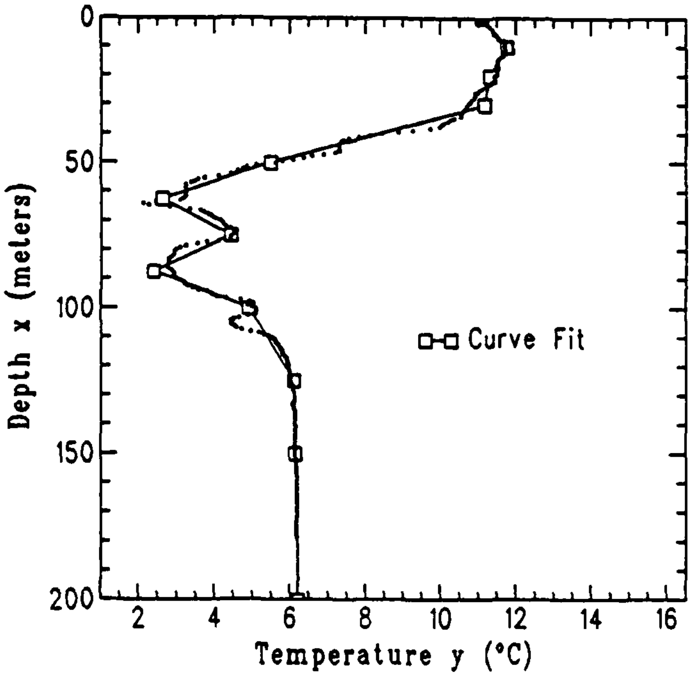


Figure 3.2: Example 1. The Least Squares Piecewise Linear Curve.

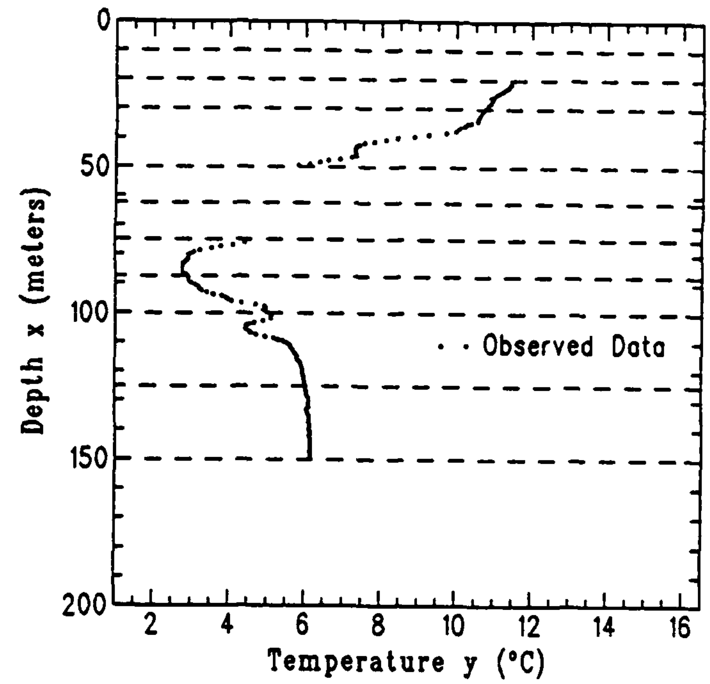


Figure 3.3: Example 2. Empty Intervals.

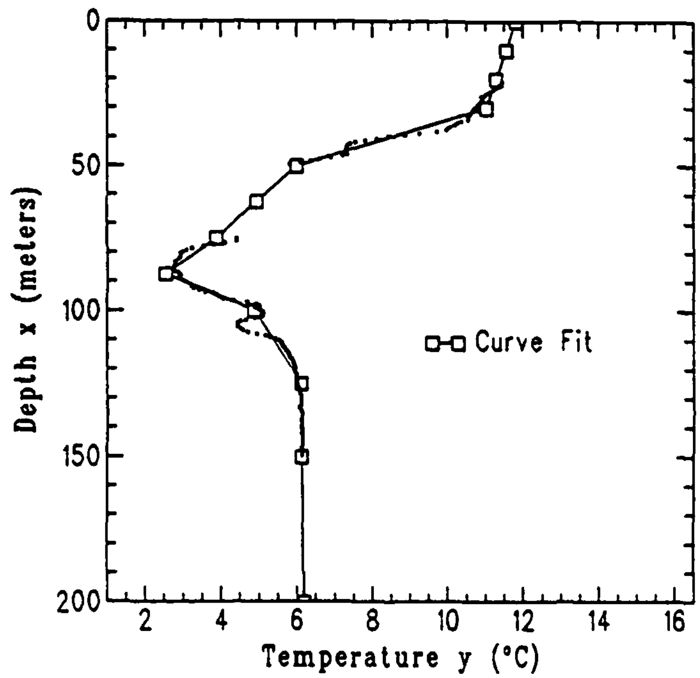


Figure 3.4: Example 2. The Least Squares Solution decomposes into two independent subproblems.

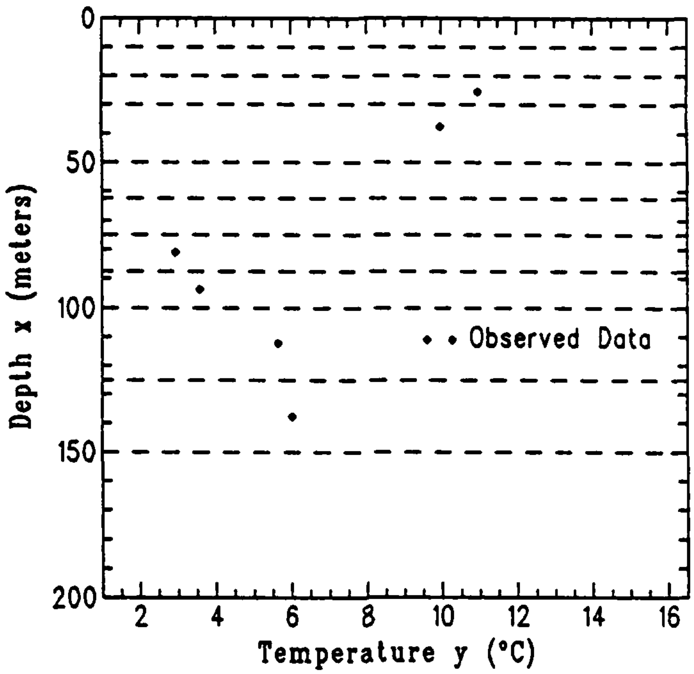


Figure 3.5: Example 3. Sparse Data.

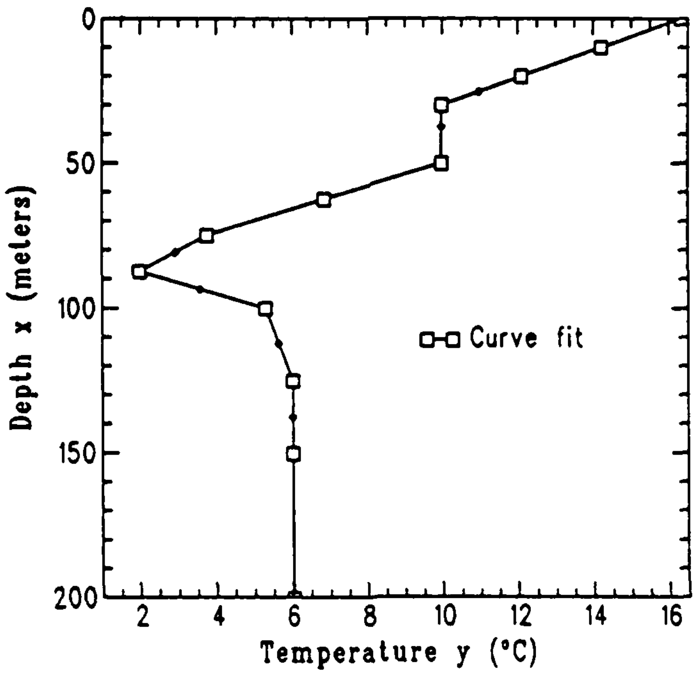


Figure 3.6: Example 3 - Solution a. Solution of a degenerate problem allows free function increment variables to be set to arbitrary user-defined values. In this case the free variables (temperature increments in intervals 4 and 10) are defined to be zero.

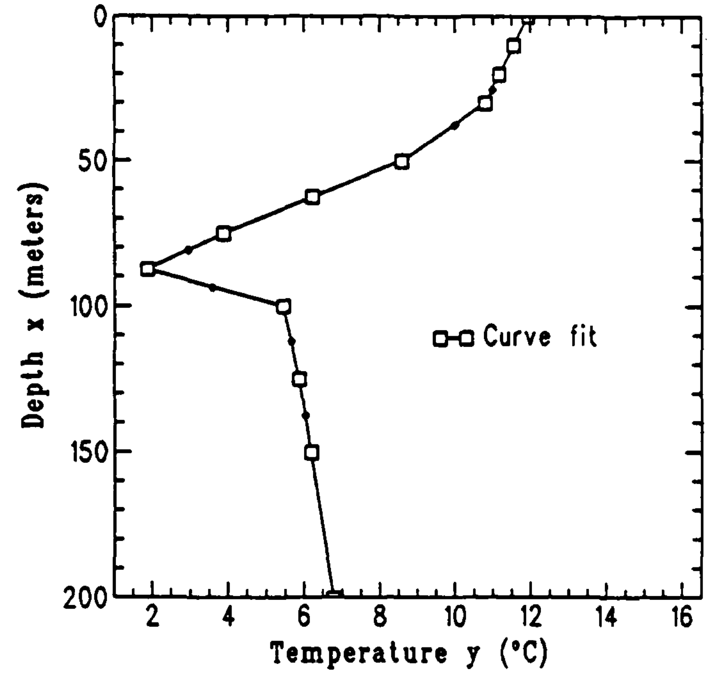


Figure 3.7: Example 3 - Solution b. The free variables (temperature increments in layers 4 and 10) are based on historical temperature gradient data.

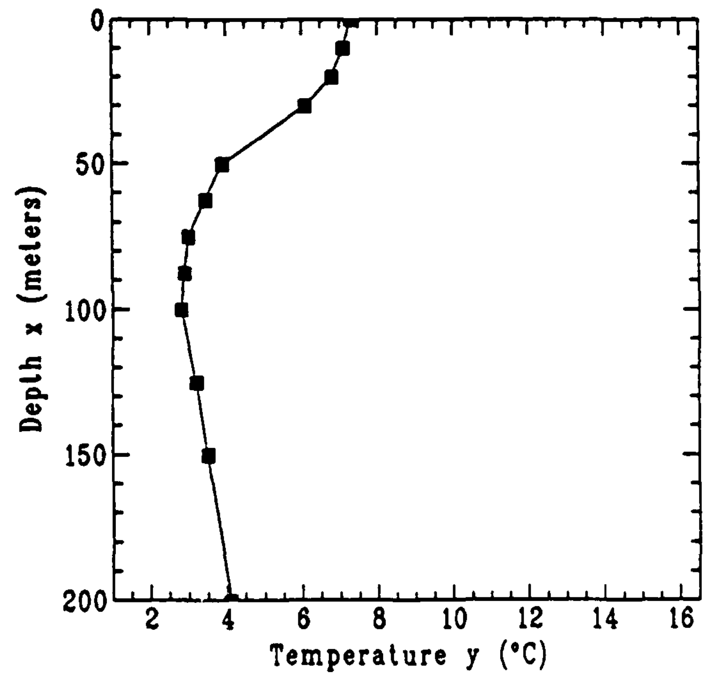


Figure 3.8: Historical temperature data. The depths define the values u_l to be used in the curve fit. The temperature increments are used as default values for undefined variables in Example 3-Solution b.

4 Summary

An algorithm has been presented for obtaining a weighted least-squares fit of a piecewise linear model to observed data, when the linear segment domains are specified. This algorithm may be used to reduce large quantities of data to more manageable size, since J coordinates are input and $L << J$ coordinates are output. The algorithm is reasonably robust to problems of data dropout, but should be used cautiously. Caution is recommended because sparse data in one segment may have undue influence on the curve fit unless suitable weighting functions are defined.

Appendix A Listing of FORTRAN Program FITBT

```
      PROGRAM FITBT
C     Fit a Piecewise Linear Curve to Bathythermal Data (Depth vs. Temperature)
C                                 or to Sound Profile Data (Depth vs. Sound Speed)
C     using Least Squares Algorithm LSQPL and Linear Equation solver SOLV.
C
C     Up to 100 unknowns (99 Linear Segments)
C*****************************************************************************
C     Input observed data from disk file LSQIN.DAT
C        J (Number of data points)
C        Y1, X1    (TEMPERATURE OR SPEED Y1, DEPTH X1)
C        ......
C        YJ, XJ    (TEMPERATURE OR SPEED YJ, DEPTH XJ)
C     No internal memory limit on the number of data points.
C*****************************************************************************
C     Optional input of historical data (up to 100 points) from disk file
C      SPRFO.DAT
C      used to define the number of segments, L, the fixed depths U(I),
C      and default values for function increment variables DELV(I)
C        (22-CHARACTER IDENTIFIER), L
C        U(1),TEMP(1),SPEED(1)  (HISTORICAL DEPTH, TEMPERATURE, SOUND SPEED)
C        .................
C        U(L),TEMP(L),SPEED(L)
C*****************************************************************************
C     Output fitted data on disk file FITBT.DAT
C        L (Number of fitted points L<<J)
C        V(1), U(1)  (TEMPERATURE OR SPEED V(1), DEPTH U(1))
C        ......
C        V(L), U(L)  (TEMPERATURE OR SPEED V(L), DEPTH U(L))
C*****************************************************************************
      CHARACTER UNITS, ID(22)
      DIMENSION UU(101)
      DIMENSION U(101),V(101),DELV(100)
      DIMENSION A(101),B(102),C(101,102),NL(100),INDEX(101),JNDEX(101)
      DIMENSION TEMP(101),SPEED(101)

C STANDARD DEPTHS IN METERS FOR DEEP WATER BATHYTHERMAL PROFILE
```

```
C TWO ADDITIONAL DEPTHS (62.5,87.5) HAVE BEEN ADDED FOR SHALLOW WATER
      DATA UU/
     1 0.0,10.,20.,30.,50.,62.5,75.,87.5,100.,125.,150.,200.,250.,300.,
     2 400.,500.,600.,700.,800.,900.,1000.,1100.,1200.,1300.,1400.,
     3 1500.,1750.,2000.,2500.,3000.,4000.,5000.,69*0./

C CONST = CONSTANT TO BE USED IN WEIGHTED ERROR FUNCTION.  CONST>=0.
C         THE WEIGHT ON THE ERROR ASSOCIATED WITH OBSERVATION JJ
C          = 1/(CONST + NO. OF POINTS IN THE SAME INTERVAL AS DATA POINT JJ)
      CONST = 10.

      LMAX = 100
      LMAX1= LMAX+1
      LMAX2= LMAX+2
      L = 30

      DO 400 I=1,LMAX1
400   U(I)=UU(I)

      WRITE(5,500)
500   FORMAT(1X'PROGRAM FITBT'/
     1  1X'VERSION 850708 - DREA - B.A.TRENHOLM'/
     2  1X'OBSERVED DATA FROM DISK FILE LSQIN.DAT: '/
     3  1X'   IN ENGLISH UNITS(0) OR METRIC(1) ? : ',$)
      READ(5,*)IFTM
      UNITS= 'ft'
      IF(IFTM.NE.0)UNITS= 'm'

      WRITE(5,501)
501   FORMAT(
     1  1X'   TEMPERATURE DATA(0) OR SOUND SPEED DATA(1) ? : ',$)
      READ(5,*)IBTVEL

C *** DEFINE VALUES DELV(I) TO CONTROL DEFAULT VALUES FOR
C      FUNCTION INCREASE IN LAYER (I) IF EQUATIONS ARE UNDERDETERMINED.
C        U(L+1)=U(L)+(U(L)-U(1))/FLOAT(L-1)
      DO 502 I=1,L
C        DEFAULT TO ZERO TEMPERATURE GRADIENT
         DELV(I)=0.0
C        DEFAULT TO SOUND SPEED PRESSURE GRADIENT
         IF(IBTVEL.NE.0)DELV(I)=0.017*(U(I+1)-U(I))
502   CONTINUE
```

```
      INFTM=1
C *** OPTION TO INPUT HISTORICAL DATA
      WRITE(5,504)
504   FORMAT(
     1 1X'READ HISTORICAL DATA FROM FILE SPRFO.DAT (0=NO, 1=YES): ',$)
      READ(5,*)IHIST
      IF(IHIST.EQ.0)GO TO 550
      WRITE(5,505)
505   FORMAT(1X'    ENGLISH UNITS (0) OR METRIC (1) ? : ',$)
      READ(5,*)INFTM
         OPEN(UNIT=23,DEVICE='DSK',FILE='SPRFO.DAT',ACCESS='SEQIN',
     1        MODE='ASCII')
         READ(23,2300)ID,L
2300     FORMAT(22A1,I5)
         IF(L.GT.LMAX)WRITE(5,506)L,LMAX
506      FORMAT(1X'L='I4,'.  MORE THAN 'I4,' SEGMENTS!')
         IF(L.GT.LMAX)STOP
         DO 507 I=1,L
         READ(23,*)U(I),TEMP(I),SPEED(I)
          IF(I.EQ.1)GO TO 507
          IF(U(I).LE.U(I-1))WRITE(5,508)I,U(I)
508       FORMAT(1X'DEPTH'I4,'='E11.5,' IS NOT IN INCREASING ORDER')
          IF(U(I).LE.U(I-1))STOP
          DELV(I-1)=TEMP(I)-TEMP(I-1)
          IF(IBTVEL.NE.0)DELV(I-1)=SPEED(I)-SPEED(I-1)
507      CONTINUE
          U(L+1)=U(L)+(U(L)-U(1))/FLOAT(L-1)
          DELV(L)=DELV(L-1)*(U(L+1)-U(L))/(U(L)-U(L-1))


C *** CHECK TO SEE IF UNIT CONVERSION IS REQUIRED
550   IF(INFTM.EQ.IFTM)GO TO 559

      IF(INFTM.EQ.0)GO TO 540
C *** CONVERT DEPTHS AND DEFAULT VARIABLES TO ENGLISH UNITS
      DO 539 I=1,L
      U(I)=U(I)*3.28084
      IF(IBTVEL.NE.0)DELV(I)=DELV(I)*3.28084
      IF(IBTVEL.EQ.0)DELV(I)=DELV(I)*9./5.
539   CONTINUE
      GO TO 559

C *** CONVERT DEPTHS AND DEFAULT VARIABLES TO METRIC UNITS
```

```
540   DO 549 I=1,L
      U(I)=U(I)/3.28084
      IF(IBTVEL.NE.0)DELV(I)=DELV(I)/3.28084
      IF(IBTVEL.EQ.0)DELV(I)=DELV(I)*5./9.
549   CONTINUE

C *** DEFINE MAXIMUM DEPTH IN FITTED PROFILE
559   INFTM=IFTM
      WRITE(5,503)UNITS
503   FORMAT(1X'INPUT WATER DEPTH ('A2,') : ',$)
      READ(5,*)WD

      DO 600 I=1,L
      IF(WD.LE.U(I))GO TO 601
600   CONTINUE
      I = L+1
601   L = I
      U(L) = WD

C *** IF THERE ARE NO DATA POINTS IN THE FIRST SEGMENTS, USE LINEAR
C     EXTRAPOLATION TO DEFINE COORDINATE VALUES
      ISETA=0

C *** IF THERE ARE NO DATA POINTS IN THE LAST SEGMENTS, USE LINEAR
C     EXTRAPOLATION TO DEFINE COORDINATE VALUES
      ISETB=0
C   * HOWEVER, IF HISTORICAL DATA ARE AVAILABLE, USE DELV(I) TO
C     OBTAIN COORDINATE VALUES
      IF(IHIST.NE.0)ISETB=1

C *** IF THE CURVE FIT PROBLEM IS UNDERDEFINED (MORE VARIABLES THAN
C     DATA POINTS) USE DELV(I) TO DEFINE FREE VARIABLES.
      ISET=1

C *** OPTIONAL DEBUG OUTPUT TO TERMINAL
      WRITE(5,5029)
5029  FORMAT(1X'OPTIONAL DEBUG OUTPUT TO TERMINAL (0=NO, 1=YES) ? ',$)
      READ(5,*)IBUG

C ***  CALL SUBROUTINE TO SET UP EQUATIONS AND SOLVE FOR UNKNOWNS
      CALL LSQPL(LMAX,LMAX1,LMAX2,L,A,B,C,NL,INDEX,JNDEX,ISET,
     1 ISETA,ISETB,U,V,DELV,N,IICK,CONST,IBUG)
```

```
      WRITE(5,5030)(I,NL(I),I=1,L)
5030  FORMAT(1X'LAYER #'I3,' CONTAINS'I5,' POINTS')

      WRITE(5,5010)N
5010  FORMAT(
     2  1XI3,'SUBPROBLEMS IDENTIFIED'/)

      IF(IICK.NE.0)WRITE(5,5020)IICK
5020  FORMAT(1XI3'SUBPROBLEMS WERE INCONSISTENT !')

      IF(IICK.EQ.0)
     1  OPEN(UNIT=22,DEVICE='DSK',FILE='FITBT.DAT',ACCESS='SEQOUT',
     2 MODE='ASCII')

      IF(IICK.EQ.0)WRITE(5,5040)
5040  FORMAT(1X'COORDINATES OF THE FITTED CURVE ARE ON FILE FITBT.DAT')

      IF(IICK.EQ.0)WRITE(22,*)L
      DO 220 I=1,L
      IF(IICK.EQ.0)WRITE(22,*)V(I),U(I)
220   CONTINUE

      IF(IICK.EQ.0) CLOSE(UNIT=22)

      STOP
      END
```

Appendix B Listing of FORTRAN Subroutine LSQPL

```
      SUBROUTINE LSQPL(LMAX,LMAX1,LMAX2,L,A,B,C,NL,INDEX,JNDEX,ISET,
     1 ISETA,ISETB,U,V,DELV,N,IICK,CONST,IBUG)
C
C     B.A.TRENHOLM, D.R.E.A., JULY 1985.
C     LEAST-SQUARES FIT OF PIECEWISE LINEAR FUNCTION
C**********************************************************************
C  INPUT VARIABLES:
C     LMAX = MAXIMUM NO. OF LINEAR SEGMENTS
C     LMAX1 = LMAX+1
C     LMAX2 = LMAX+2
C     L     = NO. OF LINEAR SEGMENTS TO BE FITTED TO DATA (L.LE.LMAX)
C     U(K)  = FIXED X COORDINATES U(1)<U(2)<...<U(L) DEFINING SEGMENTS
C     A,B,C,NL,INDEX,JNDEX ARE WORKING ARRAYS
C     ISET = 0 IF FREE VARIABLES ARE SET TO ZERO
C          = 1 IF FREE VARIABLES DEPEND ON THE INPUT VALUES DELV(K)
C              I.E. FREE VARIABLE A(1+K)=DELV(KMIN+K-1)
C           (FREE VARIABLES OCCUR WITHIN A SUBPROBLEM IF THERE ARE
C            MORE UNKNOWNS THAN DATA POINTS.)
C     ISETA = 0 LINEAR EXTRAPOLATION WILL BE USED TO DEFINE FIRST
C               COORDINATES IF THE FIRST SEGMENTS CONTAIN NO DATA POINTS.
C           = 1 DELV(I) WILL BE USED TO DEFINE FIRST COORDINATES
C               IF THE FIRST SEGMENTS CONTAIN NO DATA POINTS.
C     ISETB = 0 LINEAR EXTRAPOLATION WILL BE USED TO DEFINE LAST
C               COORDINATES IF THE LAST SEGMENTS CONTAIN NO DATA POINTS.
C           = 1 DELV(I) WILL BE USED TO DEFINE LAST COORDINATES
C               IF THE LAST SEGMENTS CONTAIN NO DATA POINTS.
C     DELV(I)= DEFAULT VALUES FOR DIFFERENCES V(I+1)-V(I), I=1,..L
C              USE OF DELV MAY BE RESTRICTED BY ISET,ISETA,ISETB.
C     CONST = CONSTANT TO BE USED IN THE WEIGHTED ERROR FUNCTION. CONST>=0.
C             THE WEIGHT ASSIGNED TO THE ERROR ASSOCIATED WITH OBSERVATION JJ
C               = 1/(CONST + NO. OF POINTS IN SAME INTERVAL AS DATA POINT JJ).
C
C     IBUG  = 0 TO SUPPRESS DEBUG OUTPUT TO TERMINAL
C           = 1 FOR DEBUG OUTPUT TO TERMINAL
C
C  INPUT FROM DISK FILE LSQIN.DAT:
```

```
C     J      = NUMBER OF DATA POINTS
C     Y1,X1 = Y AND X COORDINATES OF DATA POINT 1
C       ...
C     YJ,XJ = Y AND X COORDINATES OF DATA POINT J
C*********************************************************************
C  OUTPUT VARIABLES:
C     NL(K) = NUMBER OF DATA POINTS IN INTERVAL K, WHERE
C                INTERVAL 1 = (-INFINITY,U(2))
C                INTERVAL K = [U(K),U(K+1))
C                INTERVAL L = [U(L),+INFINITY)
C     V(K)  = VALUE OF FITTED CURVE AT U(K)
C     N     = NUMBER OF SUBPROBLEMS IDENTIFIED, EACH HAVING CONSECUTIVE
C             NONEMPTY INTERVALS
C     IICK  = NUMBER OF INFEASIBLE SUBPROBLEMS
C*********************************************************************
C  INTERMEDIATE WORKING VARIABLES FOR NTH SUBPROBLEM
C     A(1)  = INTERCEPT AT U(KMIN) OFFSET BY FO, I.E. (V(KMIN)=FO +A(1))
C     A(K)  = FUNCTION INCREMENT IN SEGMENT (KMIN + K-2), K=2,..,L(N)+1
C           = V(KMIN + K-1)-V(KMIN + K-2)
C     B     = EXTENDED ROW VECTOR FOR EQUATION CORRESPONDING TO XJ,YJ
C     C     = AUGMENTED LEAST SQUARES MATRIX WITH R.H.S. VECTOR IN LAST COL.
C     INDEX = INDEX VECTOR TO MAP SOLUTION FROM R.H.S. OF REDUCED MATRIX
C     JNDEX = INDEX VECTOR TO IDENTIFY FREE VARIABLES OF DEGENERATE MATRIX
C             AFTER MODIFIED GAUSSIAN ELIMINATION.
C*********************************************************************

      DIMENSION A(LMAX1),B(LMAX2),C(LMAX1,LMAX2),NL(LMAX),INDEX(LMAX1)
      DIMENSION JNDEX(LMAX1)
      DIMENSION U(LMAX1),V(LMAX1),DELV(LMAX)

C ADD ONE ADDITIONAL POINT
      U(L+1) = U(L) +(U(L)-U(1))/FLOAT(L-1)

C CLEAR SOLUTION VECTOR AND COUNTER
      DO 1 I=1,L
      NL(I) = 0
1     IF(ISET.EQ.0)V(I)=0.0


C N= NUMBER OF CURRENT SUBPROBLEM
      N = 0
      IICK = 0
      KLAST = 0
```

```
      KMINL = 0
      KMAXL = 0

      OPEN(UNIT=21,DEVICE='DSK',FILE='LSQIN.DAT',ACCESS='SEQIN',
     1 MODE='ASCII')

C COMPUTE AVERAGE VALUE OF DATA YJ, COUNT NUMBER OF POINTS NL IN LAYER LJ.
      READ(21,*)J
      IF(J.LE.0)RETURN

      FO = 0.0
      DO 2100 I=1,J
      READ(21,*)YJ,XJ
      XI = I
      FO = (I-1)*(FO/XI) + YJ/XI
       LJ = 1
       IF(XJ.LT.U(2))GO TO 2099
       LJ = 2
       IF(L.EQ.2)GO TO 2099
         DO 2098 K=3,L
         IF(XJ.LT.U(K))GO TO 2099
         LJ = LJ +1
2098     CONTINUE
       LJ = L
2099  NL(LJ) = NL(LJ)+1
2100  CONTINUE


2900  REWIND 21

C     TEST STATUS OF PROBLEM

C     CASE I - FUNCTION IS COMPLETELY DEFINED

      IF(KLAST.GE.L)WRITE(5,511)N
511   FORMAT(1X'SUBPROBLEM'I3, 'TERMINATES AS CASE I:ALL VARS.DEFINED')
      IF(KLAST.GE.L)CLOSE(UNIT=21)
      IF(KLAST.GE.L)RETURN

C     FIND NEXT NONEMPTY INTERVAL
      DO 100 I=KLAST+1,L
      KMIN = I
      KMAX = I
```

```
      IF(NL(KMIN).EQ.0)GO TO 100
C         NONEMPTY INTERVAL FOUND
          DO 98 K=KMIN,L
          IF(NL(K).EQ.0)GO TO 104
          KMAX=K
98        CONTINUE
          KMAX = L
          GO TO 104
100   CONTINUE

C     CASE II - REMAINING INTERVALS ARE EMPTY

      IF(ISETB.NE.0)GO TO 102
C     LINEAR EXTRAPOLATION OF REMAINING VARIABLES
      DO 101 I=KLAST+1,L
101   V(I)=V(KMAXL) + DVDU*(U(I)-U(KMAXL))

      WRITE(5,522)N
522   FORMAT(1X'SUBPROBLEM'I3,' TERMINATES AS CASE IIa:EXTRAPOLATED')
      CLOSE(UNIT=21)
      RETURN

C     REMAINING VARIABLES CONTROLLED BY INPUT VALUES DELV(I)
102   DO 103 I=KLAST+1,L
      V(I)=V(I-1)+DELV(I-1)
103   CONTINUE

      WRITE(5,523)N
523   FORMAT(1X'SUBPROBLEM'I3,' TERMINATES AS CASE IIb:CONTROLLED')
      CLOSE(UNIT=21)
      RETURN

C     CASE III - DEFINE NEXT SUBPROBLEM

104   IF(N.GT.0)WRITE(5,533)N
533   FORMAT(1X'SUBPROBLEM',I3,' COMPLETE')

      N = N + 1
C *** CLEAR LEAST-SQUARES MATRIX
      LN = KMAX-KMIN+1
      LN1 = LN+1
      LN2 = LN+2
        DO 105 I=1,LN1
```

```
      DO 105 K=1,LN2
105   C(I,K) = 0.0

C *** READ NUMBER OF POINTS IN DATA FILE
      READ(21,*)J

      DO 260 JJ=1,J
C *** READ DATA POINT JJ
      READ(21,*)YJ,XJ

C *** CHECK THAT DATA POINT IS CONTAINED IN INTERVALS FOR SUBPROBLEM N
      IF(KMAX.LT.L.AND.(XJ.GE.U(KMIN).AND.XJ.LT.U(KMAX+1)))GO TO 250
      IF(KMAX.EQ.L.AND.(XJ.GE.U(KMIN)))GO TO 250
      IF(KMIN.EQ.1.AND.XJ.LT.U(2))GO TO 250
      GO TO 260

C *** FIND LAYER LJ CONTAINING DATA POINT
250   LJ=1
      IF(XJ.LT.U(2))GO TO 258
      LJ=2
      IF(L.EQ.2)GO TO 258
          DO 259 K=3,L
          IF(XJ.LT.U(K))GO TO 258
          LJ = LJ +1
259       CONTINUE
          LJ = L
258   CONTINUE

C *** EQUATION FOR DATA POINT XJ,YJ
      DO 251 I=1,LN2
251   B(I) = 0.0
      B(1) = 1.0
      DO 252 I=1,LJ-KMIN
      B(I+1) = 1.0
252   CONTINUE
      B(LJ-KMIN+2) = (XJ-U(LJ))/(U(LJ+1)-U(LJ))
      B(LN2) = YJ-FO

C *** WEIGHTED ERROR FUNCTION
      WJ = 1.0/(CONST + FLOAT(NL(LJ)))

C *** UPDATE LEAST-SQUARES MATRIX
      DO 253 I=1,LN1
```

```
      DO 253 K=1,LN2
      C(I,K) = C(I,K) + WJ*B(I)*B(K)
253   CONTINUE

260   CONTINUE

C *** END OF DATA

C *** DEBUG PRINTOUT OF LEAST-SQUARES MATRIX
      DO 5990 I=1,LN1
      IF(IBUG.LT.0)WRITE(5,5991)I,(C(I,M),M=1,LN2)
5991  FORMAT(1X'ROW'I3,': '100E12.5)
5990  CONTINUE


C *** DEFINE DEFAULT VALUES FOR FREE VARIABLES
      IF(ISET.EQ.0)GO TO 262
      A(1)=0.0
      DO 261 I=2,LN1
261   A(I)=DELV(KMIN+I-2)
262   CONTINUE

C *** SOLVE LEAST-SQUARES EQUATIONS

      CALL SOLV(C,A,LMAX1,LN1,INDEX,JNDEX,ISET,ICK,IBUG)

C *** COUNT THE NUMBER OF INFEASIBLE PROBLEMS
      IICK = IICK + ICK
      IF(IBUG.NE.0.AND.ICK.EQ.1)WRITE(5,5992)N
5992  FORMAT(1X'SUBPROBLEM'I3,' IS INCONSISTENT')

C *** DEFINE COORDINATES V(I) OF FITTED FUNCTION AT U(I)
      IF(N.GT.1)GO TO 6002

C *** FIRST SUBPROBLEM N=1, DEFINE COORDINATES OF LEADING EMPTY INTERVALS
      IF(KMIN.EQ.1)GO TO 6002
        IF(ISETA.NE.0)GO TO 6000
C   *   LINEAR EXTRAPOLATION
        WRITE(5,513)
513     FORMAT(1X'INITIAL COORDINATES OBTAINED FROM EXTRAPOLATION')
        DO 6001 I=1,KMIN-1
        V(I) = FO + A(1) +A(2)*(U(I)-U(KMIN))/(U(KMIN+1)-U(KMIN))
6001    CONTINUE
```

```
      GO TO 6002

C   *   DETERMINED BY DELV(I)
6000    WRITE(5,512)
512     FORMAT(1X'INITIAL COORDINATES CONTROLLED')
        DO 6005 I=1,KMIN-1
        II=KMIN-I
        V(II)=V(II+1)-DELV(II)
6005    CONTINUE

C *** SUBPROBLEM N, COORDINATES OBTAINED FROM LEAST-SQUARES SOLUTION
6002  DO 6004 I=KMIN,KMAX+1
      IF(I.GT.L)GO TO 6004
      V(I) = FO + A(1)
        IF(I.EQ.KMIN)GO TO 6004
        DO 6004 K=KMIN,(I-1)
        V(I) = V(I) + A(2+K-KMIN)
6004  CONTINUE

C *** LINEAR INTERPOLATION BETWEEN SUBPROBLEMS
      IF(N.EQ.1)GO TO 6009
      DVDU =(V(KMIN)-VLAST)/(U(KMIN)-ULAST)
      DO 6007 I=KLAST+1,KMIN-1
      V(I)=VLAST+ DVDU*(U(I)-ULAST)
6007  CONTINUE

C *** STORE ENDPOINT OF SUBPROBLEM N
6009  VLAST = V(KMAX+1)
      ULAST = U(KMAX+1)
      KLAST = KMAX+1
      KMINL = KMIN
      KMAXL = KMAX

C *** PREPARE FOR LINEAR EXTRAPOLATION OF REMAINING COORDINATES
      IF(KLAST.EQ.L)GO TO 2900
      DVDU = (VLAST-V(KMAX))/(ULAST-U(KMAX))
      GO TO 2900

      END
```

Appendix C Listing of FORTRAN Subroutine SOLV

```
      SUBROUTINE SOLV(A,X,MAXN,N,INDX,JNDX,ISET,ICK,IBUG)
C       B. A. TRENHOLM, DREA, JULY 1985.
C      SOLVES N LINEAR EQUATIONS IN N UNKNOWNS
C      REDUNDANT EQUATIONS PERMITTED
C      DEFAULT VALUES MAY BE ASSIGNED TO ALL VARIABLES
C      ONE R.H.S. COLUMN
      DIMENSION A(MAXN,1),X(MAXN),INDX(MAXN),JNDX(MAXN)
C      NOTE THAT THE USER MASTER SEGMENT MUST CONTAIN THE STATEMENT
C            DIMENSION A(MAXN,M),X(MAXN),INDX(MAXN),JNDX(MAXN)
C      WHERE M IS GREATER THAN OR EQUAL TO N+1, AND WHERE N .LE. MAXN

C      A.X = LAST COLUMN OF MATRIX A
C
C      A(I,1).X(1) + ... + A(I,N).X(N) = A(I,N+1)
C
C                        FOR I = 1,...,N
C
C      INDX IS A WORKING SPACE
C      JNDX IS A WORKING SPACE
C      ISET = 0 IF FREE VARIABLES ARE DEFINED TO BE ZERO
C           = 1 IF FREE VARIABLES DEFAULT TO THE  VALUES INPUT IN VECTOR X
C             (FREE VARIABLES WILL OCCUR IF THERE ARE REDUNDANT
C              EQUATIONS OR IF THE SYSTEM IS VERY ILL-CONDITIONED.)
C      ICK = 0 IF A SOLUTION X IS FOUND
C          = 1 IF EQUATIONS ARE INCONSISTENT
C            (IN WHICH CASE X IS RETURNED AS A VECTOR OF ZEROS)
C      IBUG = 0 TO SUPPRESS DEBUG PRINTOUT
C           = 1 FOR DEBUG PRINTOUT ON TERMINAL UNIT 5

C    DETERMINE FLOAT-POINT ACCURACY, E.G. EPS=1.0E-8
      ONE = 1.
      EPS = 1.
1     EPS = EPS/10.
      EPS1 = 1. + EPS
      IF(EPS1.GT.ONE)GO TO 1
```

```
      EPS = EPS*10.
      IF(IBUG.NE.0)WRITE(5,500)EPS
500   FORMAT(1X'EPS ='E12.5)
      NPLUS1 = N+1
      DO 599 I=1,N
      IF(IBUG.NE.0)WRITE(5,598)I,(A(I,J),J=1,NPLUS1)
598   FORMAT(1X'ROW'I3,':  '100E12.5)
599   CONTINUE

C   SET INITIAL SOLUTION TO ZERO
      ICK=0
      DO 10 I=1,N
      INDX(I)=0
      JNDX(I)=0
      IF(ISET.EQ.0)X(I)=0.0
10    CONTINUE

C   NORMALIZE LINEAR EQUATIONS BY DIVIDING ALL COEFFICIENTS BY A CONSTANT
C     SO THAT THE LARGEST COEFFICIENT IS UNITY.
      EMAX = 0.0
      DO 15 I=1,N
      DO 15 J=1,N
      EMAX = AMAX1(EMAX,ABS(A(I,J)))
15    CONTINUE

      DO 16 I=1,N
      DO 16 J=1,NPLUS1
      A(I,J)=A(I,J)/EMAX
16    CONTINUE

C   SOLVE SYSTEM OF LINEAR EQUATIONS
      DO 20 J=1,N
      ZZ=EPS
C   SEARCH COLUMN J FOR LARGEST ABSOLUTE COEFFICIENT > EPS
      IF(IBUG.NE.0)WRITE(5,501)J
501   FORMAT(1X'SEARCHING COLUMN' I3)

      IROW=0
      DO 30 I=1,N
C    PREVIOUSLY ASSIGNED ROWS ARE NOT ELIGIBLE
      IF(INDX(I).NE.0) GO TO 30
      TEST=ABS(A(I,J))
      IF(TEST.LE.ZZ) GO TO 30
```

```
      ZZ=TEST
      IROW=I
30    CONTINUE

      IF(IBUG.NE.0)WRITE(5,502)IROW
502   FORMAT(1X'PIVOT ROW ='I3)

      IF(IROW.EQ.0) GO TO 20
C    PIVOT ROW FOUND
40    INDX(IROW)=J
      JNDX(J)=IROW
      ZN=A(IROW,J)
      IF(IBUG.NE.0)WRITE(5,504)ZN
504   FORMAT(1X'PIVOT ELEMENT='E11.5)
      N1=N+1
C    NORMALIZE PIVOT ROW
      DO 50 K=1,N1
50    A(IROW,K)=A(IROW,K)/ZN

C    SWEEP OUT OTHER ROWS, FOR COLUMNS > J
C      (FOR ALL COLUMNS, IF ISET NOT EQUAL TO ZERO)
      DO 66 I=1,N
      IF(I.EQ.IROW) GO TO 66
      SCALE=A(I,J)
      IF(SCALE.EQ.0.0) GO TO 66
      J1=J+1
      IF(ISET.NE.0)J1=1
      DO 60 K=J1,N1
      A(I,K)=A(I,K)-SCALE*A(IROW,K)
60    CONTINUE
66    CONTINUE
20    CONTINUE

C  CHECK TO SEE IF UNASSIGNED ROWS HAVE R.H.S. APPROXIMATELY ZERO
      DO 80 I=1,N
      IF(INDX(I).GT.0) GO TO 80
      TEST=ABS(A(I,N1))
      IF(TEST.GT.(EPS*10.))GO TO 99
80    CONTINUE

C  MOVE FREE VARIABLES TO R.H.S. OF SYSTEM
      DO 81 J=1,N
      IF(JNDX(J).GT.0)GO TO 81
```

```
            DO 81 I=1,N
            A(I,N1)=A(I,N1)-X(J)*A(I,J)
81     CONTINUE

C  PUT SOLUTION VARIABLES IN R.H.S. COLUMN INTO VECTOR X
       DO 70 I=1,N
       IF(INDX(I).EQ.0) GO TO 70
       X(INDX(I))=A(I,N1)
70     CONTINUE
       RETURN

C  NO SOLUTION
99     ICK=1
       IF(IBUG.NE.0)WRITE(5,505)I,A(I,N1)
505    FORMAT(1X'UNASSIGNED ROW #'I4,' HAS NONZERO R.H.S ='E11.5)
C      INCONSISTENT EQUATIONS
       RETURN
       END
```

Appendix D Sample Input and Output

D.1 Terminal Session for Problem 3(a)

```
 DREA Tops-20 Command processor 5.1(1712)-6
@RUN FITBT
PROGRAM FITBT
VERSION 850708 - DREA - B.A.TRENHOLM
OBSERVED DATA FROM DISK FILE LSQIN.DAT:
   IN ENGLISH UNITS(0) OR METRIC(1) ? : 1
   TEMPERATURE DATA(0) OR SOUND SPEED DATA(1) ? : 0
READ HISTORICAL DATA FROM FILE SPRFO.DAT (0=NO, 1=YES): 0
INPUT WATER DEPTH ( m) : 200
OPTIONAL DEBUG OUTPUT TO TERMINAL (0=NO, 1=YES) ? 0
INITIAL COORDINATES OBTAINED FROM EXTRAPOLATION
SUBPROBLEM  1 COMPLETE
SUBPROBLEM  2 TERMINATES AS CASE IIa:EXTRAPOLATED
LAYER #  1 CONTAINS    0 POINTS
LAYER #  2 CONTAINS    0 POINTS
LAYER #  3 CONTAINS    1 POINTS
LAYER #  4 CONTAINS    1 POINTS
LAYER #  5 CONTAINS    0 POINTS
LAYER #  6 CONTAINS    0 POINTS
LAYER #  7 CONTAINS    1 POINTS
LAYER #  8 CONTAINS    1 POINTS
LAYER #  9 CONTAINS    1 POINTS
LAYER # 10 CONTAINS    1 POINTS
LAYER # 11 CONTAINS    0 POINTS
LAYER # 12 CONTAINS    0 POINTS
  2SUBPROBLEMS IDENTIFIED

COORDINATES OF THE FITTED CURVE ARE ON FILE FITBT.DAT
CPU time 0.77   Elapsed time 17.06
```

D.2 Terminal Session for Problem 3(b)

```
@RUN FITBT
PROGRAM FITBT
VERSION 850708 - DREA - B.A.TRENHOLM
```

```
OBSERVED DATA FROM DISK FILE LSQIN.DAT:
   IN ENGLISH UNITS(0) OR METRIC(1) ? : 1
   TEMPERATURE DATA(0) OR SOUND SPEED DATA(1) ? : 0
READ HISTORICAL DATA FROM FILE SPRFO.DAT (0=NO, 1=YES): 1
    ENGLISH UNITS (0) OR METRIC (1) ? : 0
INPUT WATER DEPTH ( m) : 200
OPTIONAL DEBUG OUTPUT TO TERMINAL (0=NO, 1=YES) ? 0
INITIAL COORDINATES OBTAINED FROM EXTRAPOLATION
SUBPROBLEM  1 COMPLETE
SUBPROBLEM  2 TERMINATES AS CASE IIb:CONTROLLED
LAYER #  1 CONTAINS    0 POINTS
LAYER #  2 CONTAINS    0 POINTS
LAYER #  3 CONTAINS    1 POINTS
LAYER #  4 CONTAINS    1 POINTS
LAYER #  5 CONTAINS    0 POINTS
LAYER #  6 CONTAINS    0 POINTS
LAYER #  7 CONTAINS    1 POINTS
LAYER #  8 CONTAINS    1 POINTS
LAYER #  9 CONTAINS    1 POINTS
LAYER # 10 CONTAINS    1 POINTS
LAYER # 11 CONTAINS    0 POINTS
LAYER # 12 CONTAINS    0 POINTS
  2SUBPROBLEMS IDENTIFIED

COORDINATES OF THE FITTED CURVE ARE ON FILE FITBT.DAT
CPU time 0.89   Elapsed time 14.78
```

D.3 Input File LSQIN.DAT: Observed Data

```
 6
10.967   25.3
 9.971   37.4
 2.938   80.8
 3.582   93.5
 5.653  112.1
 6.024  137.4
```

D.4 Input File SPRFO.DAT: Historical Data

```
311085   4600N   5030W   12
     0.000         45.140      4849.069
    32.808         44.780      4847.032
```

65.617	44.240	4843.685
98.425	42.980	4835.486
164.042	39.020	4808.088
205.053	38.210	4803.349
246.063	37.400	4798.609
287.074	37.220	4798.313
328.084	37.040	4798.017
410.105	37.760	4805.905
492.126	38.300	4811.899
656.168	39.380	4823.798

D.5 Output File FITBT.DAT for Problem 3(a)

```
12
16.32845, 0.0000000E+00
14.20930, 10.00000
12.09015, 20.00000
9.971000, 30.00000
9.971000, 50.00000
6.864165, 62.50000
3.757331, 75.00000
1.991531, 87.50000
5.305008, 100.0000
6.024000, 125.0000
6.024000, 150.0000
6.024000, 200.0000
```

D.6 Output File FITBT.DAT for Problem 3(b)

```
12
11.94670, 0.0000000E+00
11.55947, 9.999878
11.17223, 20.00006
10.78500, 29.99994
8.585004, 50.00000
6.226873, 62.50015
3.868800, 75.00000
1.862740, 87.50015
5.444580, 100.0000
5.875200, 125.0000
6.175200, 150.0000
```

Bibliography

[1] G. Hadley: "Linear Programming", Addison-Wesley Series in Industrial Management, Addison-Wesley Publishing Co., Inc., New York, 1963.

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA – R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATING ACTIVITY DEFENCE RESEARCH ESTABLISHMENT ATLANTIC	2a. DOCUMENT SECURITY CLASSIFICATION UNCLASSIFIED 2b. GROUP

3. DOCUMENT TITLE
A LEAST SQUARES ALGORITHM FOR FITTING PIECEWISE LINEAR FUNCTIONS ON FIXED DOMAINS

4. DESCRIPTIVE NOTES (Type of report and inclusive dates) TECHNICAL MEMORANDUM

5. AUTHOR(S) (Last name, first name, middle initial)

TRENHOLM, BARBARA A.

6. DOCUMENT DATE September 1985	7a. TOTAL NO. OF PAGES 39	7b. NO. OF REFS 1

8a. PROJECT OR GRANT NO.	9a. ORIGINATOR'S DOCUMENT NUMBER(S) TECHNICAL MEMORANDUM 85/215
8b. CONTRACT NO.	9b. OTHER DOCUMENT NO.(S) (Any other numbers that may be assigned this document)

10. DISTRIBUTION STATEMENT

11. SUPPLEMENTARY NOTES	12. SPONSORING ACTIVITY

13. ABSTRACT

A least squared error algorithm is presented for fitting a piecewise linear function to observed data, when the slope of the function is allowed to change only at specified points. The algorithm can be used to estimate piecewise linear trends in data, where the locations of possible changes in trend are known. Furthermore, it can be used to reduce large quantities of data to manageable sizes, since J coordinates are input and $L \ll J$ coordinates are output. In addition, the algorithm is robust to problems of data dropout provided it is used cautiously. An example of a possible application is the fitting of a linear segment bathythermal profile (temperature vs. depth) to a large quantity of data.

DSIS
70-070

KEY WORDS

Curve Fitting
Linear Regression
Algorithms
Trends
Bathythermographs
Numerical Analysis
Regression Analysis
Least Squares Method
Curvilinear Regression

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the organization issuing the document.

2a. DOCUMENT SECURITY CLASSIFICATION: Enter the overall security classification of the document including special warning terms whenever applicable.

2b. GROUP: Enter security reclassification group number. The three groups are defined in Appendix 'M' of the DRB Security Regulations.

3. DOCUMENT TITLE: Enter the complete document title in all capital letters. Titles in all cases should be unclassified. If a sufficiently descriptive title cannot be selected without classification, show title classification with the usual one-capital-letter abbreviation in parentheses immediately following the title.

4. DESCRIPTIVE NOTES: Enter the category of document, e.g. technical report, technical note or technical letter. If appropriate, enter the type of document, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the document. Enter last name, first name, middle initial. If military, show rank. The name of the principal author is an absolute minimum requirement.

6. DOCUMENT DATE: Enter the date (month, year) of Establishment approval for publication of the document.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the document.

8a. PROJECT OR GRANT NUMBER: If appropriate, enter the applicable research and development project or grant number under which the document was written.

8b. CONTRACT NUMBER: If appropriate, enter the applicable number under which the document was written.

9a. ORIGINATOR'S DOCUMENT NUMBER(S): Enter the official document number by which the document will be identified and controlled by the originating activity. This number must be unique to this document.

9b. OTHER DOCUMENT NUMBER(S): If the document has been assigned any other document numbers (either by the originator or by the sponsor), also enter this number(s).

10. DISTRIBUTION STATEMENT: Enter any limitations on further dissemination of the document, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this document from their defence documentation center."

(2) "Announcement and dissemination of this document is not authorized without prior approval from originating activity."

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document, even though it may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall end with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (TS), (S), (C), (R), or (U).

The length of the abstract should be limited to 20 single-spaced standard typewritten lines; 7½ inches long.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a document and could be helpful in cataloging the document. Key words should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context.

END

FILMED

1-86

DTIC